いろいろな関数の例

山本昌志*

2006 年 10 月 25 日

概 要

C 言語の関数が使えるように練習を行う．あわせて，数学の関数をグラフ化する方法とファイル処理の基礎を学ぶ．

1 前回の復習と本日の学習内容

1.1 復習

先週は，C 言語の関数とは何か?—ということをはじめに学習した．C 言語の関数は，**処理の手続きをまとめたプログラムの単位**と説明した．プログラムの処理をまとめるれば，長いプログラムも分かり易く記述できる．なぜならば，次の 2 つの理由による．

- 何度も使う同じような処理はひとつ関数で記述できる．必要なときに，その関数を呼び出す．
- 機能単位毎に関数を使うと，プログラムが分かり易くなる．複雑な機械であっても部品単位で考えると，全体が分かるのと同じ．関数は，プログラムの部品である．

つぎに，この関数の作成方法を学習した．図 1 のようにソースプログラム中に記述する．関数を使うためには，(1) プロトタイプ宣言，(2) 関数の定義，(3) 関数の呼出し—の手続きが必要である．

1.2 本日の学習内容

本日の学習範囲は教科書 [1] の p.180–188 である．教科書とは別に，プリントに沿って先週よりは複雑，そして役に立つ関数の作成を試みる．本日の学習のゴールは以下のとおりである．

- 関数から関数を呼び出すプログラムが書ける．
- 引数や関数の戻り値に void 型を使った関数が書ける．
- 計算した関数の値をファイルに書き込める．そして，それを gnuplot を使って，グラフ化できる．

*独立行政法人　秋田工業高等専門学校　電気情報工学科

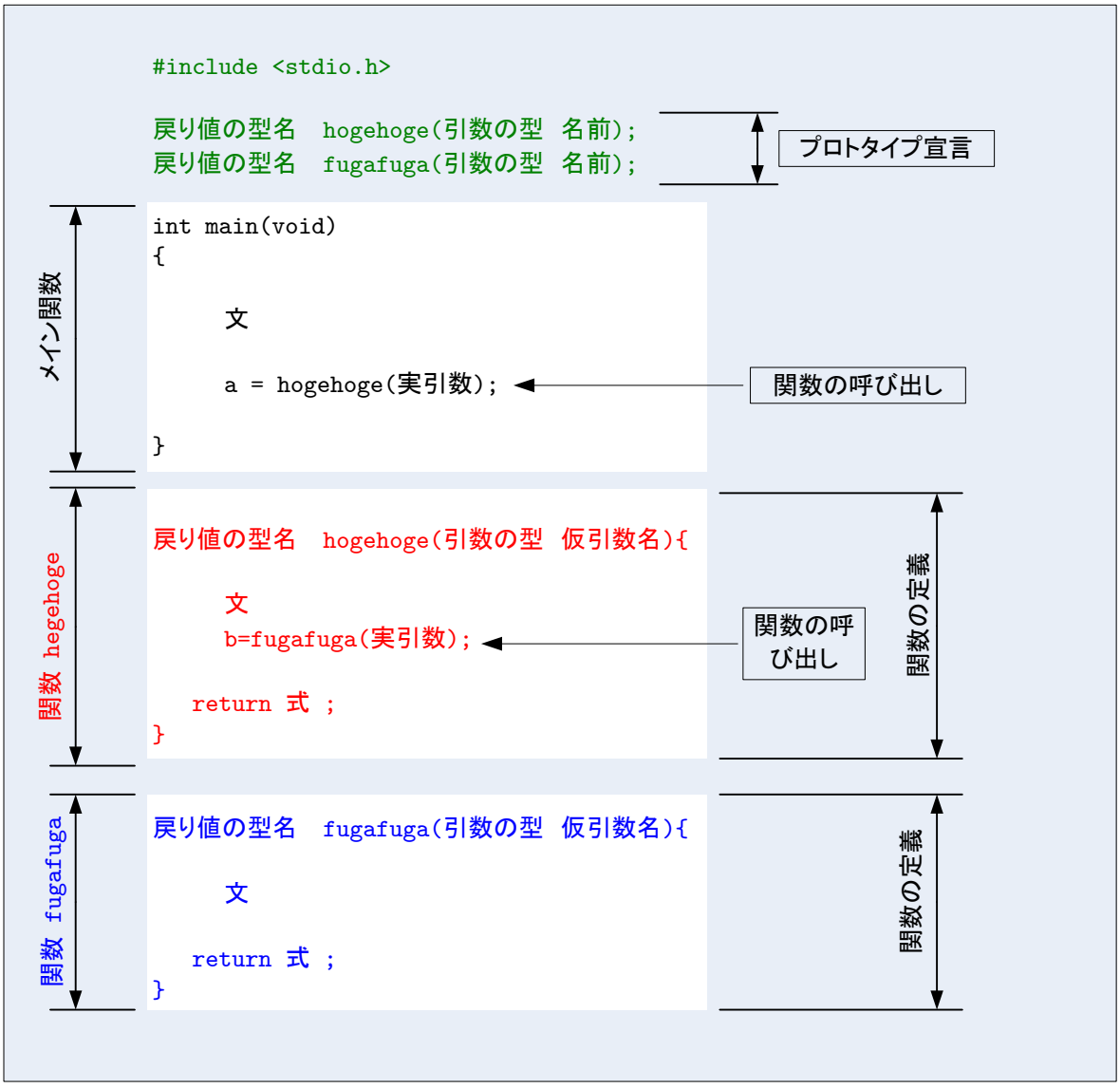

図 1: ユーザー定義関数を含んだソースプログラムの書き方

2　関数の例

2.1　関数をプロットする

ここでは，関数のグラフを描画する方法を学習する．これを身につけるとどんな関数でもグラフにすることができる．かなり便利である．

基礎数学の教科書 [2]p.82 の [例題 2] の関数

$$y = \frac{3x+2}{x+2} \tag{1}$$

をグラフに描く．この関数をグラフにするためには，次の 2 つの手順を踏む．

- x を少しずつ変化させて，そのときの y の値を計算する．このデータの組 (x, y) をファイル—ハードディスク—に保存する．
- ファイルに保存された (x, y) のデータの組を，グラフ描画ソフトウェアーでグラフ化する．

以降，この 2 つの手順について説明する．

2.1.1　関数のデータの作成

ソースプログラム　式 (1) の x の値を少しずつ変化させて，y の値を計算する—プログラムは先週の学習の範囲である．リスト 1 の `hogehoge()` 関数をループ文で繰り返し計算すればよい．

計算結果を描画するためには，x と y の値をファイルに保存する必要がある．ファイルの取扱いは，教科書 [1] の 10 章に書かれており，詳細は後に学習範する．ここでは，データの保存方法を簡単に述べる．

ファイルにデータを書き出すためには，(1) ファイル情報を格納する変数の用意，(2) ファイルのオープン，(3) データの書き出し，(4) ファイルのクローズ—という一連の動作を行う．実際には，リスト 1 のとおりで，ファイル操作に関する部分を以下に示す．

- **10 行目**　**FILE *out;**　　ファイルの情報を格納する変数宣言．`*out` がファイルの情報を格納する変数[1]と考える．`out` は `fugafuga` のようにどんな名前でも良い．しかし，先頭のアスタリスク (`*`) は必須である．
- **23 行目**　**out=fopen(”data.txt”,”w”);**　　書き込み用[2]にファイルを開いている．`data.txt` がファイル名で，`w` が書き込み用にファイルを開くことを示している．ファイル名は適当に変えてもよい．`fopen()` 関数の戻り値は，ファイル情報を格納する変数に代入する．
- **23 行目**　**fprintf(out, ”%f\t%f\n”, x, hogehoge(x));**　　ファイルにデータを書き込む．ファイルに書き込む `fprintf()` 関数は，ディスプレイに表示する `printf()` 関数とそっくりである．書き込むファイルを指定するファイル情報の変数を最初に書く—ことが異なる．
- **23 行目**　**fclose(out);**　　ファイルを閉じる．使い終わったファイルはと閉じなくてはならない．

[1] 本当はポインター．ポインターについては教科書の 8 章
[2] ファイルは，書き込みと読み込みができる．

リスト 1: 数学関数の x と y を計算し，ファイルに保存するプログラム．

```
1  #include <stdio.h>
2
3  double hogehoge(double x);                              // プロトタイプ宣言
4
5  //=========================================================================
6  // メイン関数
7  //=========================================================================
8  int main(void){
9
10   FILE *out;                                            // ファイルの情報を格納
11   int i, np;
12   double x, dx, xmax, xmin;
13
14   printf("xの最小値\t");                                 // キーボード入力
15   scanf("%lf", &xmin);
16   printf("xの最大値\t");
17   scanf("%lf", &xmax);
18   printf("プロット点の数\t");
19   scanf("%d", &np);
20
21   dx = (xmax - xmin)/np;                                //データ間隔計算
22
23   out=fopen("data.txt","w");                            //ファイルをオープン
24
25   for(i=0; i<=np; i++){
26     x = xmin + i*dx;
27     fprintf(out, "%f\t%f\n", x, hogehoge(x));           //データの書き込み
28   }
29
30   fclose(out);                                          //ファイルをクローズ
31
32   return 0;
33 }
34
35 //=========================================================================
36 //  プロットする数学関数
37 //=========================================================================
38 double hogehoge(double x){
39   double y;
40
41   y=(3*x+2)/(x+2);
42
43   return y;
44 }
```

実行と結果　このプログラムをコンパイルして，次のように実行する．

実行結果

```
xの最小値        -20
xの最大値        20
プロット点の数   10000
```

このように実行すると，ファイル「data.txt」ができる.「`emacs data.txt`」とタイプして，ファイルの中身を見ると次にようになっている．第一列目が x，第二列目が y の値である．10001 個の (x, y) の組ができている．このデータの組を線で結べば，グラフができあがる．グラフ描画方法については，次節でのべる．

出来上がったファイル (data.txt) の中身

```
-20.000000      3.222222
-19.996000      3.222272
-19.992000      3.222321
-19.988000      3.222370
-19.984000      3.222420
-19.980000      3.222469
-19.976000      3.222519
-19.972000      3.222568
-19.968000      3.222618
-19.964000      3.222668
-19.960000      3.222717
-19.956000      3.222767
-19.952000      3.222816
-19.948000      3.222866
-19.944000      3.222916

この辺は長いので，途中省略

19.980000       2.818016
19.984000       2.818049
19.988000       2.818083
19.992000       2.818116
19.996000       2.818149
20.000000       2.818182
```

2.1.2 データのプロット

データの組 (x, y) がファイルに保存されている場合，それをプロットしてグラフ化することは容易である．様々なアプリケーションプログラムがある．表計算ソフトウェアー Excel でも可能であるが，私は美しいグラフが書ける gnuplot[3] を使うことが多い．

gnuplot を使って，先ほど作成したファイルの (x, y) データを図 2 のようにグラフ化してみよう．手順は，以下のとおりである．ただし，コマンドプロンプトである`$`や`gnuplot>`はタイプしない．

gnuplot によるデータのプロット

```
$ gnuplot
gnuplot> plot [-20:20] [-50:50] "data.txt" with line
```

最初に，gnuplot を立ち上げている．すると，コマンドプロンプトが`gnuplot>`に変わる．これは，gnuplot のコマンド入力待ちであることを示している．そこで，コマンド「`plot`」でグラフを描画する．`plot` 以降は，次のような意味がある．

- `[-20:20] [-50:50]` は，x と y 軸の範囲を示している．前者が x 軸，後者が y 軸である．
- `"data.txt"`は，プロットするデータのあるファイル名である．
- `with line` は，データを直線でむすぶことを示している．

これで，ファイルに保存されたデータをグラフ化できる．gnuplot を終わりたいときには，コマンド「`exit`」をタイプする．

[3]「ニュープロット」あるいは「ヌープロット」と読む．間違いであるが「グニュープロット」と呼ばれることもある

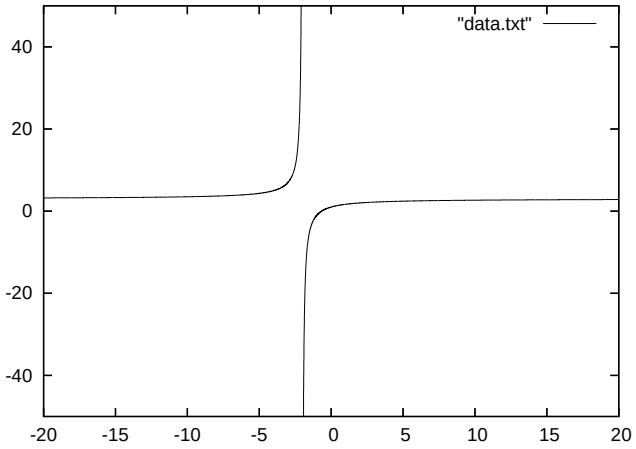

図 2: ユーザー定義関数を含んだソースプログラムの書き方

2.2 関数から関数を呼び出す

C 言語のプログラムでは，関数はいくつでも作ることができる．そして，どこからでも呼び出すことができる．複数の関数を使い，関数から関数を呼び出す例をつぎに示す．

少しばかり複雑な関数

$$y = (x^2 + \sqrt{x} + x + \sin x + \cos x)(x^2 - \sqrt{x \sin x} - 2x - \sin x + 4\cos x) + \frac{x^2 + \sqrt{x} + x + \sin x + \cos x + \tan x}{x^2 - \sqrt{x \sin x} - 2x - \sin x + 4\cos x + \tan x} \tag{2}$$

を計算することを考える．よくみると

$$y = f(x)g(x) + \frac{f(x) + \tan x}{g(x) + \tan x} \tag{3}$$

$$f(x) = x^2 + \sqrt{x} + x + \sin x + \cos x \tag{4}$$

$$g(x) = x^2 - \sqrt{x \sin x} - 2x - \sin x + 4\cos x \tag{5}$$

と書き直すことができる．この方が，すっきりしてわかりやすい．このような 3 つの数学関数は，次のようにプログラム中で記述することができる．関数 `function()` の中から，関数 `f()` と `g()` を呼び出している．

```
//=============================================================================
//  数学関数
//=============================================================================
double function(double x){
  double y;

  y=f(x)*g(x)+(f(x)+tan(x))/(g(x)+tan(x));

  return y;
}
```

```
//=========================================================================
//  f(x)
//=========================================================================
double f(double x){
  double y;

  y = x*x + sqrt(x) + x  + sin(x) + cos(x);

  return y;
}

//=========================================================================
//  g(x)
//=========================================================================
double g(double x){
  double y;

  y = x*x - sqrt(x*sin(x)) - 2*x  - sin(x) + 4*cos(x);

  return y;
}
```

2.3 void 型の関数

引数あるいは戻り値が無いときは，void 型を指定する．英語の void には,「からの, 空虚な,」という意味がある．引数や戻り値は無いが関数は手続きなので，実行すると効果はある．例えばリスト 2 の場合である．このプログラムの関数 write_file() は，(1) ファイルを開き，(2) データを書き込み，(3) ファイルを閉じる—という動作を行う．

リスト 2: void 型の関数の例.

```
1  #include <stdio.h>
2
3  void write_file(void);                          // プロトタイプ宣言
4
5  //=========================================================================
6  // メイン関数
7  //=========================================================================
8  int main(void){
9
10   write_file();                                 // 関数の呼出
11
12   return 0;
13 }
14
15 //=========================================================================
16 // void型の関数
17 //=========================================================================
18 void write_file(void){
19   FILE *fuga;
20
21   fuga = fopen("hello.txt", "w");               // ファイルのオープン
22
23   fprintf(fuga, "Hello World !!!\n");           // ファイルへの書き出し
24
25   fclose(fuga);                                 // ファイルのクローズ
26 }
```

3　プログラム作成の練習

[**練習 1**]　次の数学関数のグラフをかけ．

$$f(x) = -(x-1)^4 + 2$$

[**練習 2**]　次の数学関数のグラフをかけ．

$$f(x) = -(1-x)/(x+1)$$

[**練習 3**]　次の数学関数のグラフをかけ．ただし，$0 \leqq x \leqq 3/2$ とする．

$$f(x) = \frac{x^2 + \sqrt{x\cos x} + x}{x + \sin x + \cos x + \tan x} + (x^2 + \sqrt{x\cos x} + x + \sin x)(x + \sin x + \cos x + \tan x + x^2)$$

4　課題

次回の講義の日 (11 月 10 日) の AM8:45 までに，以下の課題をレポートとして提出すること．表紙等は，いつもの通り．表紙のタイトルは「いろいろな関数の例」とすること．

[**問 1**]　(復) 教科書の p.162–188 を 3 回読め．重要なところには，赤線あるいは蛍光ペンで印を付けよ．不明な点があれば，クラスの者に聞け．それでも分からない場合は，Office hours を利用して私 (山本) に質問しろ．

[**問 2**]　本日，配布したプリントを 3 回読め．

[**問 3**]　(復) 次の数学関数の (x, y) の値をファイルに書き込むプログラムを作成せよ．

$$f(x) = \frac{x}{x^2+1}$$

[**問 4**]　(予) 教科書の p.188–202 を 3 回読め．

[**問 5**]　(予) 以下の意味を調べ，2〜3 行程度にまとめよ．

- 自動変数
- 外部変数
- static 変数
- auto 変数
- register 変数

参考文献

[1] 内田智史監修, (株) システム計画研究所編. C 言語によるプログラミング　基礎編　第 2 版. (株) オーム社, 2006.

[2] 高遠節夫, 斎藤斉, 他. 改訂　基礎数学. 大日本印刷, 2004.